

Forum:[Hack Forums](#)

Thread: [Compiling C# 7 code at runtime](#)

severos - July 9, 2017, 8:01 am

I want to compile C# files at run time, however those files contain C#7 features -currently ValueTuples, no idea what for future-, now I thought normal code compilation works fine... however it doesn't seem so.

Current code:

PHP Code

```
<?php
namespace TestCSharp7Feature
{
    class Program
    {
        static void Main(string[] args)
        {
            var provider = new CSharpCodeProvider(new Dictionary<string, string> {{"CompilerVersion"
, "v4.0"}}});
            var options = new CompilerParameters(new[] {"System.dll", "System.Core.dll", "test.dll", true
});
            options.GenerateInMemory = false;
            options.ReferencedAssemblies.Add(Environment.CurrentDirectory +
"\\System.ValueTuple.dll");
            var res = provider.CompileAssemblyFromFile(options, "./CSharp7Feature.cs");
        }
    }
}
```

As you can see, I've already added the ValueTuple dll file since it's a nuget package.

However, when I try to run the code, it can't compile at runtime, giving me error complaining about syntax, when I check the res.Output, it looks like the code is using C#5 compiler...

Now, I've left automation for a bit and tried to manually compile using csc.exe (found in ProgramFiles MSBuild) and it gives error as well complaining about the syntax.

However, when I try to compile the file normally using VisualStudio, it just compiles and run without errors/warnings (ReSharper still complains though).

Any idea how to fix that???

Update:

As I said, compiling using csc.exe doesn't work, same goes for compiling outside VisualStudio using MSBuild.exe

However, if I use VisualStudio developer prompt I can compile using MSBuild.exe but not using csc.exe

Ninja - July 9, 2017, 8:40 am

Why would you even use tuples? They can easily be avoided.

Also, I have very little knowledge on the subject, but maybe you need to update some Nuget packages? Try "Microsoft.Net.Compilers", and "Microsoft.CodeDom.Providers.DotNetCompilerPlatform".

severos - July 9, 2017, 9:01 am

(Jul 9, 2017 08:40 AM)Ninja Wrote: [\[CITE\]](#) Why would you even use tuples? They can easily be avoided.

Also, I have very little knowledge on the subject, but maybe you need to update some Nuget packages? Try "Microsoft.Net.Compilers", and "Microsoft.CodeDom.Providers.DotNetCompilerPlatform".

I have no idea why using tuples, it's one just a requirement for work, been struggling for like 5 hours before thinking of asking here XD

I've already updated everything, also as I said in the update, building works using MSBuild.exe from VS dev prompt only, others keep failing because they don't recognize the syntax for tuple destruction regardless of referenced dll...

Washi - July 9, 2017, 10:00 am

Tuples in C# 7 are amazing, especially tuple deconstruction, which allows you to immediately use the individual components of a tuple as variables:

```
PHP Code
<?php
public (int x, bool y) MyMethod() {
//...
}

var (left, right) = MyMethod();
Console.WriteLine(left);
Console.WriteLine(right);
```

I mean how can you not like that ?

On topic: As far as I know, the CodeDom namespace does not support C# 6 or higher. What you need is the compiler service provided by [Roslyn](#). There are various nuget packages out there that act as a replacement. One example being: <https://www.nuget.org/packages/Microsoft.CodeDom.Providers.DotNetCompilerPlatform/> . Or use the newest version of MSBuild, which is now not shipped with the .NET Framework by default anymore and, when installed, is located in Program Files instead.

Ninja - July 10, 2017, 9:15 am

(Jul 9, 2017 10:00 AM)Washi Wrote: [\[CITE\]](#) Tuples in C# 7 are amazing, especially tuple deconstruction, which allows you to immediately use the individual components of a tuple as variables:

```
PHP Code
<?php
public (int x, bool y) MyMethod() {
//...
}

var (left, right) = MyMethod();
Console.WriteLine(left);
```

Console.WriteLine(right);

I mean how can you not like that ?What about the overhead? I heard it is significantly less, so is it 'good' to use now?

severos - July 10, 2017, 11:41 am

(Jul 10, 2017 09:15 AM)Ninja Wrote: [\[CITE\]](#) What about the overhead? I heard it is significantly less, so is it 'good' to use now?

Far as I read, the main reason to use tuples is if you want to return multiple values, they are slower when passing around since they're structs, but faster in allocation (GC too? don't remember tbh), so if you want to hold values for very short amount of time (like in returning multiple values from function) tuples have better performance than classes, but if you're going to move it around much then classes are better.

Washi - July 10, 2017, 11:48 am

(Jul 10, 2017 09:15 AM)Ninja Wrote: [\[CITE\]](#) What about the overhead? I heard it is significantly less, so is it 'good' to use now?

Tuples in C# 7 are represented by the [ValueTuple structure](#). Because it is a structure a.k.a. a value type, it means that allocating any instance of it locally will just result in the compiler storing the tuple onto the stack in its entirety, instead of allocating it onto the heap and storing a pointer onto the stack. This is the case when using an instance of the Tuple class (or any instance of a reference type).

I have never really tested it though if it is actually faster or not, but it would not surprise me at all because of the above reason. I am pretty sure though that it does slow down when you keep on using the tuples to pass values on, as it needs to copy the entire data every time it is returned or passed on. In that case, an object would be better.

severos - July 11, 2017, 5:10 am

Ok, I kinda solved this, after messing around with VS dev prompt I was able to find the exact command for csc.exe to compile c#7 code, the whole problem was that the "default" libraries included weren't the required ones, so I had to specifically reference them. What I did was write the files I want to compile into a directory (for easier command) and made a batch file then run it. could have done without batch file, but passing arguments became too annoying so writing to file was easier.